

InterSymp-2000, Baden-Baden, Germany, Jul.31-Aug.4, 2000. *KYM-IS00*

# An Inference Engine-Based Subscription Service

By

Kym Jason Pohl

Collaborative Agent Design (CAD) Research Center

California Polytechnic University

San Luis Obispo, California, USA

A Distributed Network Architecture (DNA) forms the basis of many decision-support systems available today. Such architecture excels at providing an application environment rich in transparency, performance, and expandability (Gray and Lievano 1997). However, efforts at the Collaborative Agent Design (CAD) Research Center in San Luis Obispo, California have found that for this architecture to be successfully applied to agent-based, decision-support systems two additional ingredients must be present, namely a query service and a subscription service. These two mechanisms support the extensive collaboration inherent in agent-based, decision-support systems. Although primarily utilized as a means of agent-to-agent interaction these two services can be employed by any collaborator, including users.

## Query Service

Common among many information management systems the query service provides the ability for any collaborator to obtain specific data or information based on a given set of constraints. In a sense, the query service allows data or information to be *pulled* from single or multiple repositories. As such, queries exist as synchronous requests. In the classical sense currency of query results is maintained exclusively through subsequent queries. As a result, if information currency is important within the scope of a decision-support application a potentially substantial overhead must be accrued as a result of frequent query requests. Even with such sacrifice in performance it is unrealistic that a high degree of currency can consistently be maintained. This is especially true in the case of opportunistic agent-based, decision-support systems where incremental analysis is performed as additional information becomes available.

## Subscription Service

One solution to the issue of information currency is the use of a subscription service. A subscription service essentially monitors a dynamic set of client interests or subscriptions. In this sense, a subscription can be thought of as a *standing* or continuous query. Similar to individual queries, interests can be described in terms of information values and constrained events. Once posted, these interests are monitored by the subscription service on a continuous basis. If satisfied the subscription service notifies all interested parties of the situation. This notification can even be coupled with the *pushing* of contextual event information to appropriate subscribers. However, in many cases the relevant information a subscriber seeks is that the event has occurred and the subscriber may not in fact be concerned with actual event context. In this case it is more efficient to

decouple the notification of interest satisfaction and the conveyance of contextual information. If desired such information can be obtained through a series of follow-on queries issued by the subscriber.

## Ontological System

Resulting from several years of developing decision-support systems of this nature it has been the experience of the CAD Research Center that the effectiveness of both a query service and subscription service can be significantly increased if the underlying information these services operate on is represented as an ontological system. An ontology is defined as a collection of objects, object characteristics, and inter-object relationships describing a particular domain (Chandrasekaran et al. 1999). These domains may range from system-level concepts (e.g., application management, information management, etc.) to specific characteristics regarding the domain of application (e.g., military mission planning, crisis management, engineering design, etc.). Regardless of focus, such representation allows both constrained queries and subscriptions to be explicitly described in terms of objects and object characteristics.

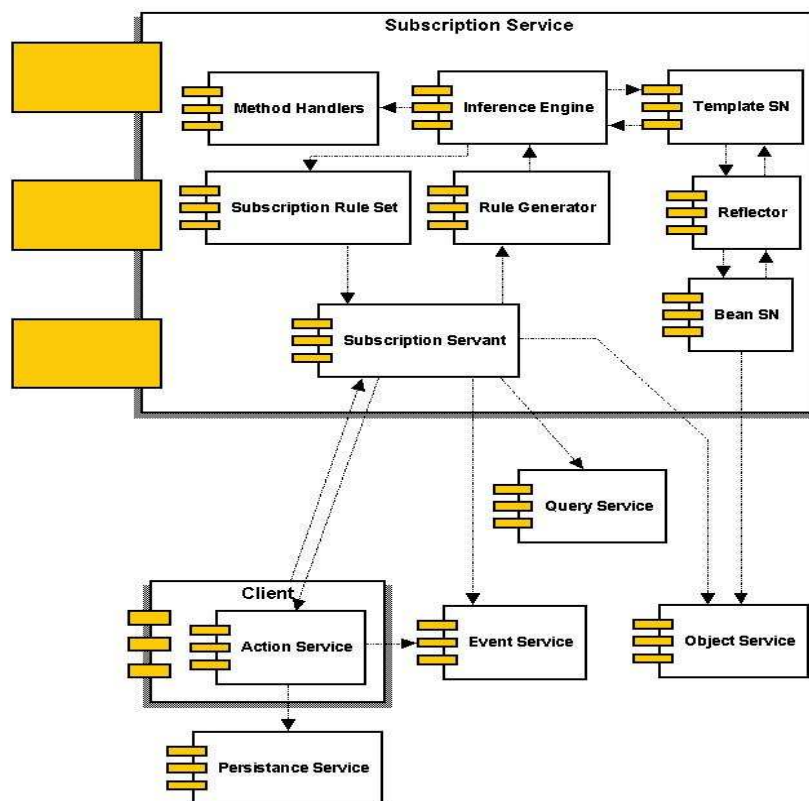
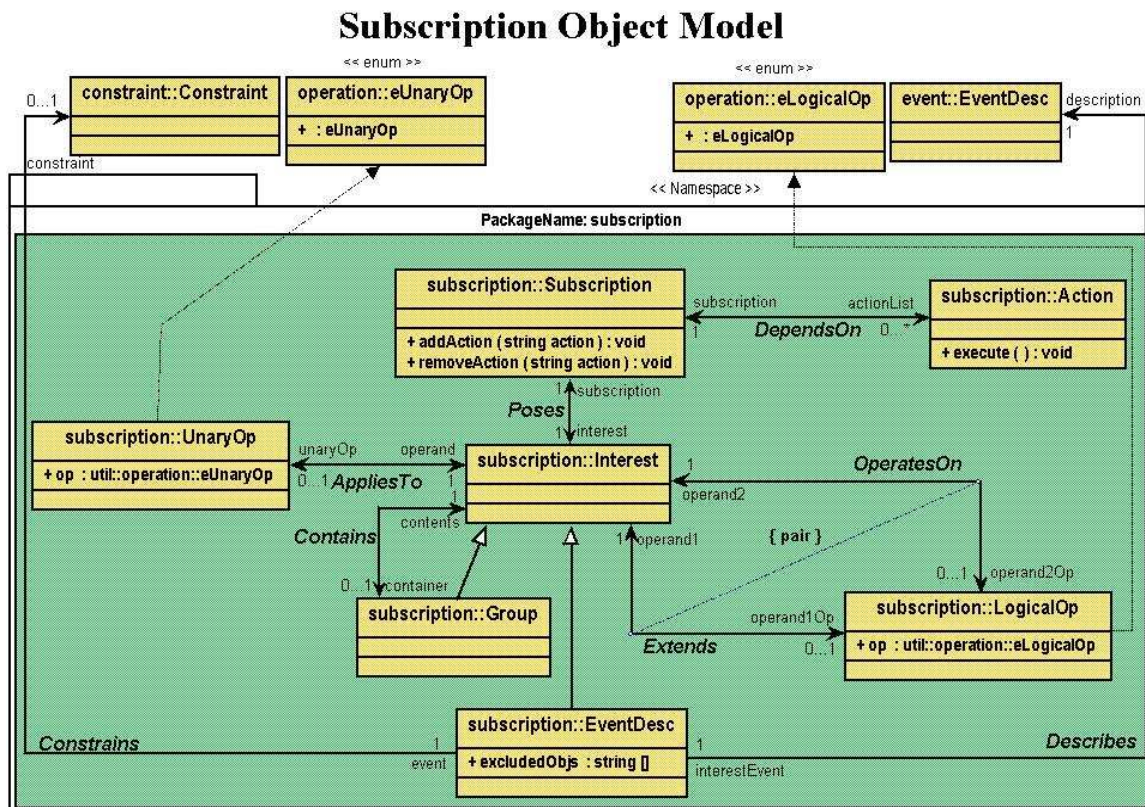


Figure 1 – Inference engine-based subscription service architecture

## Inference Engine

An inference engine essentially manages the satisfaction of predicate logic based on a dynamic set of knowledge (Weiss 1999). Such mechanisms have and continue to be an effective basis for many decision-support systems (Pohl et al. 1997). The subscription service implementation presented in this paper focuses on the CLIPS rule-based inference engine developed by NASA (NASA 1992). In the case of CLIPS, predicate logic is described in terms of rules, potentially complex patterns together with related actions. Each rule has a pattern describing a set of conditions and a subsequent action to execute upon its satisfaction. To efficiently manage the matching of potentially high volumes of patterns across equally high and ever-changing pools of information CLIPS employs an extremely efficient scheme known as the rete algorithm (Forgy 1982). As a result the varying degree of pattern satisfaction across large sets of rules and dynamic sets of information can effectively be maintained in the context of a near real-time decision-support system (Pohl 1997). It is this efficient and extensive pattern matching ability that makes this mechanism an excellent candidate for forming the core of a robust, ontology-based subscription service.

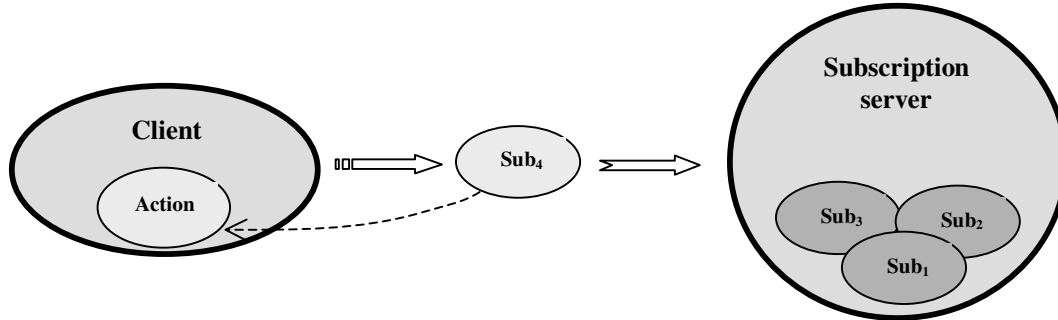


**Figure 2 – Subscription service domain ontology described**

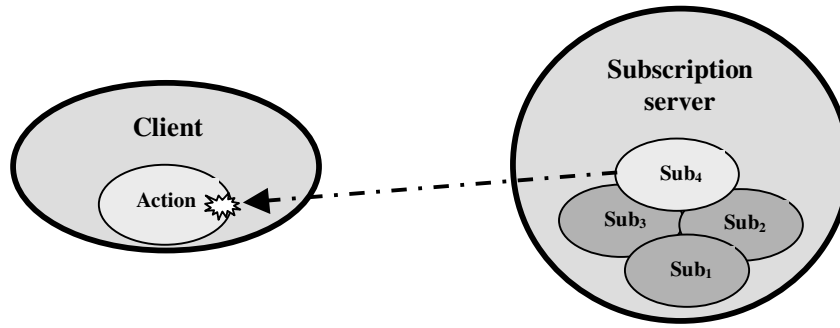
## Inference-Based Subscription Service

Figure 1 illustrates the basic components comprising the subscription service architecture. Each component works in conjunction with the others to effectively manage the dynamic set of subscriptions. Implemented within a CORBA-based environment (Mowbray and Zahavi 1995) this architecture adheres closely to the application server design pattern (Ayers et al. 1999). In such a pattern both information and functionality are essentially *served* to a dynamic set of clients as sharable, collaborative objects. Following this pattern, the subscription service is presented to clients as instantiatable Subscription objects embodying the same set of qualities as any CORBA object. Employing an identical representational approach as the aforementioned agent-based, decision-support systems developed by the CAD Research Center these objects adhere to an ontology, in this case, outlining the domain of subscription/notification (Figure 2).

To invoke the subscription service a client may either instantiate a subscription object or instead chose to utilize an existing subscription registered by another client (i.e., subscription objects enjoy the CORBA quality of being sharable). Regardless of method, during this process the subscriber also associates a local action object to the subscription (Figure 3a). It is this action that the client wishes to have executed upon subscription satisfaction. Each time a new subscription is created the Rule Generator component of the subscription service constructs a corresponding CLIPS rule. The content of this rule represents the characteristics of the particular subscription. This rule is then placed under the management of the CLIPS inference engine that in turn monitors the pattern for satisfaction. If a match occurs the action portion of the rule (not to be confused with the client action object) triggers the associated client action. Like subscription objects, client action objects exist as specialized CORBA objects. However, while subscription objects are implemented within the scope of the subscription service, action objects are implemented within the context of the subscribing client. This introduces a powerful capability inherent in CORBA-based architectures. In a CORBA-based paradigm the distinction between client and server is somewhat blurred. That is, each application component has the potential of being a client in one sense and a server in another. The notification mechanism outlined in this design makes significant use of this feature to permit asynchronous communication between the subscription service and its clientele. Once the subscription service identifies that an interest has indeed been satisfied the subscription service then becomes a client to the action server part of the subscriber it intends to notify. In a CORBA-like fashion, the subscription service remotely executes the *notify* method of the action object of each relevant client (Figure 3b). Once they are executing within the context of their action objects, notified subscribers may determine the most appropriate course of action to react to the event. Recall from the previous discussion that such action may involve utilizing the query service to obtain more detailed, contextual information regarding the event.



**Figure 3a – Subscription registration**



**Figure 3b – Client notification regarding interest satisfaction**

## Conclusion

By employing an inference engine as the core of the subscription service two significant capabilities are achieved. First, subscribers in a decision-support application can now exploit the powerful and extensive pattern matching functionality inherent in a rule-based inference engine such as CLIPS. The degree and complexity of subscriber interests are constrained only by the extent of the ontological system on which they operate.

Second, a strong similarity can be drawn between subscriptions comprised of extensive interests together with specific actions and the powerful pattern/action architecture of an expert system rule. In this manner, the subscription service outlined in this paper allows decision-support applications to be described as collections of *distributed* expert system rules thus encompassing the inherent benefits of the autonomous and opportunistic nature of such designs.

## References

Ayers D, H Bergsten, M Bogovich, J Diamond, M Ferris, M Fleury, A Halberstadt, P Houle, P Mohseni, A Patzer, R Phillips, S Li, K Vedati, M Wilcox, and S Zeiger; *"Professional Java Server Programming"*; Wrox Press, Birmingham, UK, 1999.

Chandrasekaran B, J Josephson, R Benjamins; *"What Are Ontologies, and Why Do We Need Them?"*; IEEE Intelligent Systems, January/February 1999.

Forgy C; *"Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem?"*; Artificial Intelligence, Vol 19, 1982 (pp. 17-37).

Fowler M and K Scott; *"UML Distilled: Applying the Standard Object Modeling Language"*; Addison-Wesley, Reading, Massachusetts, 1997.

Gray S and R Lievano; *"Microsoft Transaction Server 2.0"*; SAMS Publishing, Indianapolis, Indiana, 1997.

Hayes-Roth F, D Waterman and D Lenat (eds.); *"Building Expert Systems"*; Addison-Wesley, Reading, Massachusetts, 1983.

Mowbray T and R Zahavi; *"The Essential CORBA: Systems Integration Using Distributed Objects"*; John Wiley and Sons, Inc., New York, New York, CA, 1995.

NASA; *"CLIPS 6.0 Reference Manual"*; Software Technologies Branch, Lyndon B Space Center, Houston, Texas, 1992.

Orfali R, D Harkey and J Edwards; *"The Essential Distributed Objects Survival Guide"*; John Wiley and Sons, Inc., New York, New York, CA, 1996.

Pohl J, A Chapman, K Pohl, J Primrose and A Wozniak; *"Decision-Support Systems: Notions, Prototypes, and In-Use Applications"*; Technical Report, CADRU-11-97, CAD Research Center, Design Institute, College of Architecture and Environmental Design, Cal Poly, San Luis Obispo, CA, January, 1997.

Pohl J; *"Human-Computer Partnership in Decision-Support Systems: Some Design Guidelines"*; in Pohl J (ed.) *Advances in Collaborative Design and Decision-Support Systems*, focus symposium: International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany, August 18-22, 1997 (pp. 71-82).

G Weiss (ed.); *"Multiagent Systems"*; MIT Press, Cambridge, Massachusetts, 1999.